Ömer Mesud TOKER
21302479
CS 315 – 2
HW 1

Part A)

In both codes built in Python lists and NumPy arrays `otherData = data` do not create new array or list. `otherData` is just another name of `data` list.

However, in Python lists, `otherData = data[1:3]` creates new list whose name is `otherData`, and it includes the elements with indices 1 and 2 of `data` list. In NumPy arrays, `otherData = data[1:3]` does not create a new array. It is just a reference to `data`'s second indices (which is numbered as 1).

Part B)

When '*' operator is used for NumPy arrays, then result will be formed by multiplying the components element-wise.

When '*' operator is used for NumPy matrices, then result will be formed by matrix multiplication.

'**' operator is a power operation for both but for arrays it is element-wise but for matrices it is matrix multiplication.

Part D)

There are solutions to this problem. We can use NumPy arrays to solve the problem too. If the graph is sparse (there are lots of 0s in the matrix representation), then we can use another approach to model the graphs to reduce the space the graph occupies, that is, the adjacency lists. Instead of using NumPy matrices, we can use the arrays of NumPy arrays, with each array element of outer NumPy array includes different number of elements. Using this approach to sparse graphs, we can model the graph in a shorter memory space. Also, dictionary of keys, coordinate list, yale format, compressed sparse row and compressed sparse column techniques can be used. In addition, SciPy library has some functions to solve this problem.